FASTMC on UNIX plateforms with old and new LUND compatibility

V. Breton, C. Marchand, April 1994

ABSTRACT

Your favorite CLAS simulator FASTMC is now avalaible on UNIX plateforms DEC-ULTRIX, HP-UX and SUN-OS5. This new version supports input events in either old LUND (JETSET6x) or new LUND (JETSET7x) formats, and makes no longer any use of CELEG libraries, but CERN JETSET73. This note describes all changes relative to the VMS version described in CLAS-note-90-003.

1. Organization of files

The UNIX FASTMC package contains the following files:

- *source code:*

fast_main.f: fast_main and routines most likely to be changed by user (fast_anal, fast_end_user, fast_hstdef, fast_list + user routines)
fast_sources.f: all other routines, including SPLINELIB.

- *common files:*

fast_common.inc: general FASTMC common blocks
fast_lund.inc: LUND common blocks
fast_geom.inc: CLAS geometry common blocks
fast_nstar.inc: N* simulation common blocks
trig_common.inc: trigger simulation common blocks

- *input files:*

fast_input: description of general simulation parameters
trigspec: description of the trigger

The content of these files is as described in CLAS-note-90-003, except for the changes described below.

2. Changes relative to previous VMS version

Because the new version is linked with CERN JETSET73 library, all specific LUND common blocks and routine calls had to be adapted, as there are major differences between JETSET6x and JETSET7x. Compatibility with the old LUND format is maintained by duplication of the common block LUJETS. The changes are almost transparent to users which use the routine *fast_getpar* to retrieve the events, as it returns **ipart** according to the old LUND particle code scheme. All changes between the two versions are commented in the code source (labelled C. Marchand, 03/94) and are described in the subsequent sections.

2.1. *Input files* **lund** *and* **fast_input**

The events which have to be processed by FASTMC should be as usual stored unformatted in a file named **lund**. The new version of FASTMC supports both old LUND (JETSET6x) or new LUND (JETSET7x) data structure. The user specifies which one is used by adapting the logical variable OLDLU in the file **fast_input**: OLDLU = 'T' : old LUND format; OLDLU = 'F' : new LUND format. (The "fast data" structure is still supported in the old LUND convention).

2.2. *Data structure and LUND common blocks*

The structure of the events in the LUND file should respect the structure of the common block LUJETS:
old LUND: COMMON/LUJETS/n,k(2000,2),p(2000,5)
new LUND: COMMON/LUJETS/n,k(4000,5),p(4000,5),v(4000,5)

The compatibility between old and new LUND conventions is ensured by:

- duplication of the common LUJETS in the **fast_lund.inc** file
old: common/LUJETO/n,k(4000,2),p(4000,5) (for compatibility)
new: common/LUJETS/new,knew(4000,5),pnew(4000,5),v(4000,5) (official)
- conversion routines OLDNEW and NEWOLD to translate common LUJETO to LUJETS and vice versa. They make use of two useful routines KFNEW(K) and KFOLD(KNEW) to convert particle codes. These routines (in file **fast_sources.f**) are called right after event read in and the user has access to both old and new LUND data structure through the whole FASTMC routines where **fast_lund.inc** is included.

2.3. *Summary of major changes between JETSET6x and JETSET7x*

The user is urged to read the JETSET73 manual (available from E. Smith) for a full description of the changes, especially the common LUDAT1, LUDAT2... defining the infinity of switches to turn on particle decay, adapt the hadronization scheme,.... We will concentrate below on the changes affecting the common block LUJETS.

***Correspondence between LUJETO and LUJETS**

n = new = number of particles in the event

p = pnew = momentum (p(i,1),p(i,2),p(i,3)), energy(p(i,4)) and mass (p(i,5)) of the ith particle.

v(i,5): vertex information about the ith particle (new option in JETSET7x)

k(i,1) = 10000*KS + KH; k(i,2) = KF
knew(i,1) = KS; knew(i,2) = KF; knew(i,3) = KH

where KS gives information for decay, KF is the particle code and KH the rank of the parent particle for a decay product. KS and KF have changed from JETSET6x to JETSET7x, whereas KH is unchanged.

***Old and new KS convention**

Old	New	meaning
-	0	empty line
0	1	undecayed particle or unfragmented jet (last of a jet system)
1	2	unfragmented jet, followed by more jets in the same color singlet jet system
2	11	decayed particle or fragmented jet (last of a jet system)
3	12	fragmented jet, followed by more jets in the same color...
4	-	original beam or target particle

***Old and new KF convention**

part	old	new	part	old	new	part	old	new	part	old	new
γ	1	22	π^0	23	111	p	41	2212	Δ^{++}	61	2224
Z^0	2	23	η	24	221	n	42	2112	Δ^{+}	62	2214
W^+	3	24	η'	25	331	Σ^+	43	3222	Δ^{0}	63	2114
e^-	7	11	η_c	26	441	Σ^0	44	3212	Δ^{-}	64	1114
ν_e	8	12	ρ^+	27	213	Σ^-	45	3112	Σ^{*+}	65	3224
μ^-	9	13	K^{*+}	28	323	Ξ^0	46	3322	Σ^{*0}	66	3214
ν_μ	10	14	K^{*0}	29	313	Ξ^-	47	3312	Σ^{*-}	67	3114
τ^-	11	15	D^{*0}	30	423	Σ_c^{++}	48	4222	Ξ^{*0}	68	3324
ν_τ	12	16	D^{*+}	31	413	Σ_c^{+}	49	4212	Ξ^{*-}	69	3314
π^+	17	211	ρ^0	33	113	Σ_c^{0}	50	4112	Ω^-	70	3334
K^+	18	321	ω	34	223	Λ^0	57	3122	Σ_c^{*++}	71	4224
K^0	19	311	Φ	35	333	Λ_c^+	58	4122	Σ_c^{*+}	72	4214
D^0	20	421	J/Ψ	36	443				Σ_c^{*0}	73	4114
D^+	21	411	K_S^0	37	310						
F^+	22	431	K_L^0	38	130						

All antiparticles of above cited charged particles have negative codes.

JETSET7x supports also some useful new particle codes, especially for nuclei:

```
- deuteron: 811
- triton  : 821
- alpha   : 822
```

3. Where to find UNIX FASTMC and how to run it

- The UNIX FASTMC package includes all the source files described in section 1 (which are machine independent) and one machine dependent **makefile**.
It is located in: **/usr/user3/marchand/fastmc/release** for HP-UX
It is located in: **/usr/user7/marchand/fastmc/release** for DEC-ULTRIX
A makefile for SUN-OS5 is available on request.

- To install FASTMC in your directory:
cp ~marchand/fastmc/release/.* .*
make
and you will have an executable version called *fastmc*. To run it, just type *fastmc*, provided you have an event input file called **lund** in binary format. Output files are generated according to what you have specified in *fast_anal*, *fast_hstdef* and *fast_end_user*. The example file **fast_main.f** was intended for simulation of PR93-031 and generates histograms (**fast_hbk**) and ntuples (**ntuple_fast.dat**), as well as the usual **fast_list**.

GOOD LUCK!